UNSUPERVISED LEARNING OF NEURAL SEMANTIC MAPPINGS WITH THE HUNGARIAN ALGORITHM FOR COMPOSITIONAL SEMANTICS

Xiang Zhang[1] *Shizhu He*[2,*] *Kang Liu*[2] *Jun Zhao*[2]

[1]School of Artificial Intelligence, University of Chinese Academy of Sciences
[2]The Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences

ABSTRACT

Neural semantic parsing maps natural languages (NL) to equivalent formal semantics which are compositional and deduce the sentence meanings by composing smaller parts. To learn a well-defined semantics, semantic parsers must recognize small parts, which are semantic mappings between NL and semantic tokens. Attentions in recent neural models are usually explained as one-on-one semantic mappings. However, attention weights with end-to-end training are shown only weakly correlated with human-labeled mappings. Despite the usefulness, supervised mappings are expensive. We propose the unsupervised Hungarian tweaks on attentions to better model mappings. Experiments have shown our methods is competitive with the supervised approach on performance and mappings recognition, and outperform other baselines.

Index Terms— semantic parsing, hungarian algorithm, attention mechanism, compositional semantics

1. INTRODUCTION

Semantic parsing transduces natural language sentences into formal semantic representations understandable or executable for intelligence agents, such as the untyped λ-calculus, LISP and SQL. Formal semantics are compositional and complete meanings are composed by smaller components, up to atomic semantic mappings, for example "less than" being translated to $<$ operator in SQL, which may vary across domains and schemas. Thus semantic parsers have to learn two subtasks, the recognition and composition of semantic mappings.

Neural parsers are agnostic about the subtasks and trained end-to-end. Although competitive on many benchmarks, they had been shown inferior to understand compositional semantics [1]. We hypothesize the potential defect lies in the attention modules, which are designed to recognize the mappings from natural language words to semantic tokens and to produce the context vectors. The attentions produce dense weights and lack supervisions, which weaken their effects of recognizing semantic mappings and thereby increase the burden on the compositions for the model.

One straightforward solution is to make the attention weights sparser. For example, substitute the Softmax with SparseMAX [2] in the attention modules, or use temperature-annealing schedule during training. They can improve the weight sparsities significantly and are expected to reasonably solve the recognition subtask. However, due to the lack of convincing feedbacks, end-to-end training of the sparse attentions doesn't assure meaningful mappings or better performance, although their sparsities are enhanced.

Another solution is to manually label the mappings [3]. Although the consensus among human labels are not easier to obtain, the supervision significantly improve the performance and the model interpretability. However, manual labels are expensive and even questionable for complex semantics like the deeply nested SQL queries in the ATIS dataset.

To circumvent the issues, we propose the Hungarian tweak for classic attentions. We simplify the semantic mappings as an one-on-one assignment problem, i.e., forcing each word being aligned to one single token, but leaving the tokens unaligned if the words are exhausted and vise versa. In the language of the assignment problems, we shall find the maximal matching reward on a weighted bipartite graph (words against tokens), and the mappings are then considered optimal. In each iteration during training, we solve the assignment problem characterized by the runtime attention weights, and use the optimal assignments in the loss function. Similar to the EM algorithm, the training process simultaneously guides the model to update the estimated attention weights, and maximizes the attention-based reward. The simple unsupervised tweak is not only competitive to the SOTA and supervised alternative, but also shown better agreements and sparser weights to other sparse attention counterparts.

Furthermore, we find the effectiveness of Hungarian

*corresponding author

This work was supported by National Key R&D Program of China (No.2022ZD0118501) and the National Natural Science Foundation of China (No.62376270, No.U1936207, No.61976211). This work was supported by the Strategic Priority Research Program of Chinese Academy of Sciences (No.XDA27020100), Yunnan Provincial Major Science and Technology Special Plan Projects (No.202202AD080004) and OPPO Research Fund.

tweaks is depended on the dataset. Every example of SQuALL requires only one table and shows clean connections between the natural language words and the where clauses in SQL queries. However, for complex representations like the deeply nested SQL queries in the ATIS dataset, the one-on-one simplification can not be accurate. Instead of turning the problem to a more sophisticated version (like many-to-many structural mappings), we find the tweak also effective if used in reverse. Specifically, contrary to picking up the optimal assignments for the loss function, we build the negative feedbacks for attentions by excluding the least optimal mappings. The modification is also simple and shown effective on several complex SQL datasets. Note that we focus on SQL only and dismiss the more concise representations like FunQL and λ-calculus, because they show a closer proximity of syntax to natural language and are not comparable to representations like Python and SQL.

In this work we highlight the importance of semantic mappings recognition for compositional semantics. By casting the semantic mappings into an assignment problem, we propose the unsupervised Hungarian tweaks that is competitive to the supervised SOTA. And we also propose the reverse Hungarian tweak, which is easy to implement, for datasets where token-level mappings are not explicit. By evaluations on a variety of datasets, we show the attention modules trained with Hungarian tweaks can generate more accurate weights and have better interpretability.

2. METHODOLOGY

2.1. Formalization

We choose an encoder-decoder model for each dataset, follow its formalization, and only add training losses for the attention modules. Specifically, the model input is a natural language sentence $x = \{x_i\}_{i=1}^{n}$, and generates an SQL query $y = \{y_j\}_{j=1}^{m}$. If x is accompanied with a table, the table header t is concatenated, forming $x = [x; t]$. A model may have multiple attention modules. One typical implementation is adopted in the decoder f_d which attends to hidden states $H = \{h_i\}_{i=1}^{n} = f_e(x)$ produced by the encoder f_e, as Eq.1.

$$c_j = \sum_i f_{norm}(f_{score}(q_j, k_i))v_i = \sum_i w_{ij} v_i \tag{1}$$

where the keys k and values v are transformed from H, the query q_j is transformed from the decoder state at time step j, f_{norm} tries to normalize the score into weights that sum to one, and f_{score} is the score function. We keep the score functions of the baseline, such as the dot-product $f_{dp}(x, y) = x^T y$ and the bilinear $f_b(x, y) = x^T W y$ functions.

For f_{norm} the Softmax is commonly used, but we also try using the SparseMAX as suggested in Section 1. We also bring in a linear annealing schedule for comparison, which change the temperature τ in $f_\tau(\mathbf{x}) = Softmax(x/\tau)$ during training. The τ is initialized to 1 and equivalent to the standard Softmax, and decreased to 0.1 at the last epoch. The two hyperparameters are fixed and not investigated.

2.2. Hungarian Attention

We bias the attention weights by adding additional loss terms L_{hun} to the original cross-entropy loss as $L = L_{xent} + \lambda L_{hun}$. The key to our method is to cast the attentions to assignment problems. A feasible solution to the problem is a binary matrix $A = \{a_{ij} \in \{0, 1\}\}$. a_{ij} is 1 if x_i is assigned to y_j and both must not be involved in other assignments. An optimal A^* maximizes the reward $R(A)$ in Eq.2.

$$\max_A R(A) = \sum_{a_{ij} \in A} a_{ij} w_{ij} \tag{2}$$
$$s.t. \quad a_{ij} \in \{0, 1\}, \sum_j a_{ij} \leq 1, \text{ and } \sum_i a_{ij} \leq 1$$

where w_{ij} is the runtime attention weight calculated as Eq. 1. The Hungarian algorithm is one of the standard and polynomial-time algorithms to the problem. We adopt the variant one from the SciPy package to support rectangular weight matrix ($n \neq m$).

Since the algorithm implementations are not differentiable, we build two losses based on MSE and cross-entropy respectively as Eq.3 and Eq.4 to train attention parameters.

$$L_{hun}^{mse} = \sum_{a_{ij} \in A^*} \frac{(w_{ij} - a_{ij})^2}{2} \cdot \mathbb{1}[a_{ij} \neq \bar{w}_{ij}] \tag{3}$$

$$L_{hun}^{xent} = \sum_{a_{ij} \in A^*, a_{ij}=1} -\log(w_{ij}) \cdot \mathbb{1}[a_{ij} \neq \bar{w}_{ij}] \tag{4}$$

where the optimal solution is represented by the binary tensor $A^* \in \mathbb{R}^{n \times m}$. The $\mathbb{1}[\cdot]$ is the indicator function and the $\bar{w}_{ij}$ is binary and equals 1 if and only if w_{ij} is the maximal weight of the i-th row. In this way, the loss will be disabled dynamically for w_{ij} that is sufficiently good, thereby eliminating the unwanted update especially at the early stage of training when the attention weights are not well-trained. With supervised oracle mappings O, A^* in Eq. 3 can be replaced with O, yielding the supervised loss term L_{sup}.

Note the attention weight matrix w_{ij} is column-normalized by f_{norm}. It thus can tell the mapping strength over each x_i given some j, but it doesn't fit the exclusive constraint as Eq.2. The Hungarian attention can be seen as exerting constraints across the rows, biasing the weight matrix towards discrete and exclusive mappings.

2.3. Reverse Hungarian Attention

For datasets that are both small and complex, the Hungarian tweaks may not be effective because the data are clearly not reducible to simple one-on-one mappings. For example, the ATIS contains deeply nested SQL queries, a constraint may correspond to a complete subquery. We hypothesize that the

Hungarian attention can at least be used in reverse. Instead of making the mappings more sparse and explicit, we can remove the least relevant context from x for generating c_j. Although the Hungarian algorithm can solve the minimal optimization of Eq.2, we build a new weight matrix C as Eq.5,

$$c_{ij} = (1 - w_{ij})/(n - 1) \tag{5}$$

such that C is as column-normalized as W, and any pair (c_{ij}, c_{kj}) of the column $C_{\cdot j}$ satisfies that $c_{ij} < c_{kj}$ if and only if $w_{ij} > w_{kj}$. In this way, if we substitute W with C in Eq.2, the Hungarian algorithm can give the least probable mappings without changing the code and loss function. And the gradient to w will be in the reverse direction of c.

3. RESULTS ANALYSIS

Table 1 lists the dataset scales. Every example contains a natural language question x and a corresponding SQL query y. But SQuALL examples have an additional table t and the labeled mappings among x, t, and y. Other datasets assume a fixed schema and do not come with mapping labels.

Table 1. Statistics of datasets used in experiments.

	ATIS	GEO	Scholar	Advising	SQuALL
Train	3014	409	433	3440	9030
Dev.	405	103	111	451	337
Testing	402	95	105	446	4344

We combine ATIS, GEO, Scholar, and Advising into a single dataset AGSA due to their limited size, forcing one model to parse for all. We follow the i.i.d. splits [4] of AGSA, and report results with 5 random seeds. SQuALL has 5 different splits by default and we run experiments on each of them. We adopt one of the dominant models for every dataset, and only compare attention modules trained with the proposed Hungarian tweaks and the basic setup within that model. Specifically, we choose the baseline model [3] for SQuALL and the Seq2Seq baseline for AGSA.

3.1. Analysis on SQuALL

The basic model for SQuALL is a BERT encoder and an LSTM decoder with bilinear attentions. The Softmax and SparseMAX as f_{norm} are compared. Oracle mappings use the manual labels as the drop-in replacement of w_{ij}(Eq. 1). The supervised loss L_{sup} uses manual labels to improve each setting. As the unsupervised alternatives, our Hungarian tweaks use the L_{hun}^{mse} loss following the baseline model which uses MSE for L_{sup}. For Softmax we also include the annealing schedule for comparison. The performance are listed in Table 2.

Similar to [3], the Oracle mappings by human workers, have outperformed other settings, although the labels may lack consensus. In addition, we find the Oracle mappings can be further improved significantly with supervised loss L_{sup}, which means the manual labels can not only direct the alignment structures, but also enhance the token representations and network parameters. Moreover, the supervised loss can improve both Softmax and SparseMAX, indicating that they cannot yet capture the implicit mappings. Therefore, we're delighted to see the unsupervised Hungarian tweaks, could achieve similar performance against the supervised loss.

Table 2. Dev accuracies of different SQuALL splits

Setup	Dev0	Dev1	Dev2	Dev3	Dev4	Mean
Softmax	40.6	44.8	43.6	**46.9**	45.2	44.2
+ L_{sup}	43.4	**47.6**	**43.7**	45.7	46.4	**45.4**
+ annealing	38.6	41.6	39.5	40.6	44.7	41.0
+ L_{hun}	**43.5**	47.2	42.7	44.3	**46.4**	44.8
Sparsemax	36.6	40.1	35.8	35.0	39.2	37.3
+ L_{sup}	**42.7**	**46.3**	**43.6**	**44.4**	45.0	**44.4**
+ L_{hun}	42.3	44.6	40.8	43.4	**45.5**	43.3
Oracle	59.5	64.0	59.5	58.7	61.0	60.5
+ L_{sup}	**61.8**	**65.4**	**61.4**	**60.8**	**62.5**	**62.4**

Table 3. Gini Indices on Dev0 split for different setups.

	S-Q	S-T	Q-T	T-Q
Softmax	0.7685	0.8272	0.9141	0.9013
+ L_{sup}	0.4885	0.249	0.2351	0.3499
+ annealing	0.8949	0.9091	0.9648	0.9981
+ L_{hun}	0.4455	0.2368	0.1327	0.5577
Sparsemax	0.9004	0.8280	0.8295	0.901
+ L_{sup}	0.5434	0.2497	0.2506	0.4166
+ L_{hun}	0.5892	0.3023	0.2700	0.697
Oracle	0.0173	0.7409	0.7157	0.8426
+ L_{sup}	0.0178	0.7467	0.724	0.8235

Attention sparsities are analyzed in Table 3 with the Gini Index [5]. Higher Gini values indicate sparser weights. Several attentions exist in the model among **Q**uestion words, **T**able headers, and **S**QL tokens. Initial letters S, Q, and T are used to describe attentions. For example, the S-Q attention uses SQL as q and Question words as k in Eq. 1.

Recall that the Oracle is manually labeled. The learned sparsities are expected to be leaned towards the Oracle. In Table 3, the dense S-Q of Oracle is the most prominent, which means few SQL tokens would be connected to the question. There is a strong negative correlation between the S-Q Gini values and the Dev0 accuracies. The Hungarian tweaks and supervised loss have both significantly reduced the S-Q Gini, which explains their better performance. However, the SparseMax and the τ annealing boost the sparsity of S-Q, which is not required and thus harms the model.

For other attentions (S-T, Q-T, T-Q), the Oracle gives a moderately high sparsity. The softmax and sparsemax can

Table 4. Ablation accuracies of settings by removing the Hungarian tweak for every attention modules on SQuALL

	Dev0
Hungarian Attentions	43.5
- no SQL-Question	42.3
- no SQL-Table	42.8
- no Question-Table	43.0
- no Table-Question	42.2

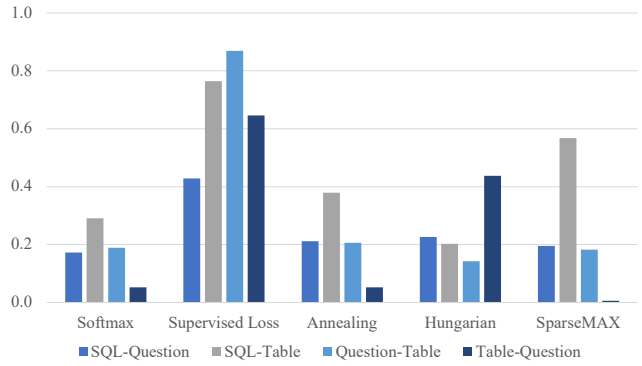

Fig. 1. Recall of the mappings on different setups. For a query q_j the key k_i with the highest weights w_{ij} is considered as the predicted mapping and compared with the Oracle.

give sparser weights than the Hungarian tweaks and even the supervised loss. We investigate the contradiction with ablations. Table 4 tells the S-Q and T-Q have greater impacts on the improvements than S-T and Q-T.

Moreover, because sparsity can not tell whether attentions have learned the correct mappings, we pick the mappings with the greatest attention weights and compare with the Oracle. Figure 1 shows the supervised loss has nonetheless learned the most correct mappings. But the Hungarian tweaks do well on both S-Q and T-Q, which are more important from Table 4 Therefore, we can conclude that Hungarian tweaks is more flexible than the end-to-end training.

3.2. Analysis on AGSA

We use Seq2Seq with bilinear attentions on AGSA, and the L_{hun}^{xent} is used for the Hungarian tweak due to its better performance against L_{hun}^{mse}. Following the convention [4] we use 5 random seeds and report the mean results for the combined AGSA dataset. Accuracies are evaluated by exact matches of the predicted SQL against the gold targets.

Table 5 compares the effect of Hungarian tweaks. As suggested in Section 2.3, the deeply nested SQL queries in AGSA are too complex that the simplification of token-level one-on-one mappings is not applicable. Although the Hungarian tweak can improve attention sparsities, the performance is significantly worse. With the SQL representations, it's also

Table 5. Accuracies with 5 random seeds on AGSA dataset.

	Dev	Test
Seq2Seq (Baseline)	71.35±1.13	69.87±0.81
+ Hungarian Attn.	65.81±1.73	64.52±0.95
+ Reverse Hungarian Attn.	**71.50**±0.80	**70.42**±1.26

Table 6. Results on AGSA dev with one fixed seed

	EM Accuracy	Gini Index
Seq2Seq (Baseline)	70.01	0.8199
+ Hungarian Attn.	64.21	**0.8757**
+ Reverse Hungarian Attn.	**71.59**	0.8535

hard to specify phrase-level mappings even manually. But with the help of reverse Hungarian tweak, where we pick up the least possible attention and build negative feedbacks, the attention weights get moderately sparser, and the performance is also improved. The implementation is lightweight to most neural models against methods utilizing span-based mappings, but also brings improvements.

4. RELATED WORK

Neural semantic parsers follows the encoder-decoder structures [6] and are trained end-to-end instead of being compositional. To reduce grammatical errors, Seq2Tree [7] is proposed to generate nested trees for untyped λ-calculus. Grammar-based decoders [8,9] are then proposed to generate sequences of rules instead of tokens, in the depth-first order of target AST. Other parsers design intermediate patterns over the targets [10–15]. The abstraction layer can be seen as handcrafted structures for the targets, which can reduce the learning difficulties for the model. Similarly, span-level mappings [16] are proposed to reduce the complexity for the source. Moreover, formalizations such as the Tree Substitution Grammar [17], and constituency boundaries [18] are also proposed to explore other grammatical decodings other than the CFGs. However, their work presumes that semantic mappings can be correctly captured by attentions, orthogonal to our goal to improve the attention training.

5. CONCLUSION

In this paper, we have addressed the importance of semantic mappings for compositional semantics. Vanilla attentions have given worse results and failed to predict the mappings. The forced sparsity by SparseMAX and annealing is not flexible and may not predict the correct mappings. The proposed Hungarian tweaks on attentions can predict better mappings as manual labels and only encourage sparsities on demand. For datasets not suitable for flat one-on-one mappings, we show the reverse Hungarian tweaks can reduce noisy mappings and increase sparsities along with the performance.

6. REFERENCES

[1] Daniel Keysers, Nathanael Schärli, rli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet, "Measuring compositional generalization: A comprehensive method on realistic data," in *Proceedings of ICLR*, 2020.

[2] Andre Martins and Ramon Astudillo, "From softmax to sparsemax: A sparse model of attention and multi-label classification," in *Proceedings of The 33rd International Conference on Machine Learning*, Maria Florina Balcan and Kilian Q. Weinberger, Eds., New York, New York, USA, 20–22 Jun 2016, vol. 48 of *Proceedings of Machine Learning Research*, pp. 1614–1623, PMLR.

[3] Tianze Shi, Chen Zhao, Jordan Boyd-Graber, Hal Daumé III, and Lillian Lee, "On the potential of lexico-logical alignments for semantic parsing to sql queries," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, Online, Nov 2020, p. 1849–1864, Association for Computational Linguistics.

[4] Inbar Oren, Jonathan Herzig, Nitish Gupta, Matt Gardner, and Jonathan Berant, "Improving compositional generalization in semantic parsing," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online, Nov. 2020, pp. 2482–2495, Association for Computational Linguistics.

[5] Niall Hurley and Scott Rickard, "Comparing measures of sparsity," *IEEE Transactions on Information Theory*, vol. 55, no. 10, pp. 4723–4741, Oct 2009.

[6] Chunyang Xiao, Marc Dymetman, and Claire Gardent, "Sequence-based structured prediction for semantic parsing," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, Aug. 2016, pp. 1341–1350, Association for Computational Linguistics.

[7] Li Dong and Mirella Lapata, "Language to logical form with neural attention," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, Aug. 2016, pp. 33–43, Association for Computational Linguistics.

[8] Pengcheng Yin and Graham Neubig, "TRANX: A transition-based neural abstract syntax parser for semantic parsing and code generation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Brussels, Belgium, Nov. 2018, pp. 7–12, Association for Computational Linguistics.

[9] Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner, "Neural semantic parsing with type constraints for semi-structured tables," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, Sept. 2017, pp. 1516–1526, Association for Computational Linguistics.

[10] Li Dong and Mirella Lapata, "Coarse-to-fine decoding for neural semantic parsing," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, July 2018, pp. 731–742, Association for Computational Linguistics.

[11] Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang, "Towards complex text-to-SQL in cross-domain database with intermediate representation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, July 2019, pp. 4524–4535, Association for Computational Linguistics.

[12] Jiwei Ding, Wei Hu, Qixin Xu, and Yuzhong Qu, "Leveraging frequent query substructures to generate formal queries for complex question answering," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, Nov. 2019, pp. 2614–2622, Association for Computational Linguistics.

[13] Bo Chen, Xianpei Han, Ben He, and Le Sun, "Learning to map frequent phrases to sub-structures of meaning representation for neural semantic parsing," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 0505, pp. 7546–7553, Apr 2020.

[14] DongHyun Choi, Myeong Cheol Shin, EungGyun Kim, and Dong Ryeol Shin, "RYANSQL: Recursively applying sketch-based slot fillings for complex text-to-SQL in cross-domain databases," *Computational Linguistics*, vol. 47, no. 2, pp. 309–332, June 2021.

[15] Lunyiu Nie, Shulin Cao, Jiaxin Shi, Jiuding Sun, Qi Tian, Lei Hou, Juanzi Li, and Jidong Zhai, "GraphQ IR: Unifying the semantic parsing of graph query languages with one intermediate representation," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, Dec. 2022, pp. 5848–5865, Association for Computational Linguistics.

[16] Pengcheng Yin, Hao Fang, Graham Neubig, Adam Pauls, Emmanouil Antonios Platanios, Yu Su, Sam Thomson, and Jacob Andreas, "Compositional generalization for neural semantic parsing via span-level supervised attention," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online, June 2021, pp. 2810–2823, Association for Computational Linguistics.

[17] Jing Zheng, Jyh-Herng Chow, Zhongnan Shen, and Peng Xu, "Grammar-based decoding for improved compositional generalization in semantic parsing," in *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada, July 2023, pp. 1399–1418, Association for Computational Linguistics.

[18] Daking Rai, Bailin Wang, Yilun Zhou, and Ziyu Yao, "Improving generalization in language model-based text-to-SQL semantic parsing: Two simple semantic boundary-based techniques," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Toronto, Canada, July 2023, pp. 150–160, Association for Computational Linguistics.